

#include <algorithm>

#include <cmath>

#include <cstdio>

using namespace std;

const int maxn = 200010;

int f[maxn], g[maxn], id[maxn], head[maxn], cnt, last[maxn], dep[maxn],

fa[maxn][22], v[maxn], w[maxn];

int block, index, n, m, q;

int pos[maxn], col[maxn], app[maxn];

bool vis[maxn];

long long ans[maxn], cur;

struct edge {

int to, nxt;

} e[maxn];

int cnt1 = 0, cnt2 = 0; // 时间戳

struct query {

int l, r, t, id;

bool operator<(const query &b) const {

return (pos[l] < pos[b.l]) || (pos[l] == pos[b.l] && pos[r] < pos[b.r]) ||

(pos[l] == pos[b.l] && pos[r] == pos[b.r] && t < b.t);

}

} a[maxn], b[maxn];

inline void addedge(int x, int y) {

e[++cnt] = (edge){y, head[x]};

head[x] = cnt;

}

void dfs(int x) {

id[f[x] = ++index] = x;

for (int i = head[x]; i; i = e[i].nxt) {

if (e[i].to != fa[x][0]) {

fa[e[i].to][0] = x;

dep[e[i].to] = dep[x] + 1;

dfs(e[i].to);

}

}

id[g[x] = ++index] = x; // 括号序

}

inline int lca(int x, int y) {

if (dep[x] < dep[y]) swap(x, y);

if (dep[x] != dep[y]) {

int dis = dep[x] - dep[y];

for (int i = 20; i >= 0; i--)

if (dis >= (1 << i)) dis -= 1 << i, x = fa[x][i];

} // 爬到同一高度

if (x == y) return x;

for (int i = 20; i >= 0; i--) {

if (fa[x][i] != fa[y][i]) x = fa[x][i], y = fa[y][i];

}

return fa[x][0];

}

inline void add(int x) {

if (vis[x])

cur -= (long long)v[col[x]] \* w[app[col[x]]--];

else

cur += (long long)v[col[x]] \* w[++app[col[x]]];

vis[x] ^= 1;

}

inline void modify(int x, int t) {

if (vis[x]) {

add(x);

col[x] = t;

add(x);

} else

col[x] = t;

} // 在时间维上移动

int main() {

scanf("%d%d%d", &n, &m, &q);

for (int i = 1; i <= m; i++) scanf("%d", &v[i]);

for (int i = 1; i <= n; i++) scanf("%d", &w[i]);

for (int i = 1; i < n; i++) {

int x, y;

scanf("%d%d", &x, &y);

addedge(x, y);

addedge(y, x);

}

for (int i = 1; i <= n; i++) {

scanf("%d", &last[i]);

col[i] = last[i];

}

dfs(1);

for (int j = 1; j <= 20; j++)

for (int i = 1; i <= n; i++)

fa[i][j] = fa[fa[i][j - 1]][j - 1]; // 预处理祖先

int block = pow(index, 2.0 / 3);

for (int i = 1; i <= index; i++) {

pos[i] = (i - 1) / block;

}

while (q--) {

int opt, x, y;

scanf("%d%d%d", &opt, &x, &y);

if (opt == 0) {

b[++cnt2].l = x;

b[cnt2].r = last[x];

last[x] = b[cnt2].t = y;

} else {

if (f[x] > f[y]) swap(x, y);

a[++cnt1] = (query){lca(x, y) == x ? f[x] : g[x], f[y], cnt2, cnt1};

}

}

sort(a + 1, a + cnt1 + 1);

int L, R, T; // 指针坐标

L = R = 0;

T = 1;

for (int i = 1; i <= cnt1; i++) {

while (T <= a[i].t) {

modify(b[T].l, b[T].t);

T++;

}

while (T > a[i].t) {

modify(b[T].l, b[T].r);

T--;

}

while (L > a[i].l) {

L--;

add(id[L]);

}

while (L < a[i].l) {

add(id[L]);

L++;

}

while (R > a[i].r) {

add(id[R]);

R--;

}

while (R < a[i].r) {

R++;

add(id[R]);

}

int x = id[L], y = id[R];

int llca = lca(x, y);

if (x != llca && y != llca) {

add(llca);

ans[a[i].id] = cur;

add(llca);

} else

ans[a[i].id] = cur;

}

for (int i = 1; i <= cnt1; i++) {

printf("%lld\n", ans[i]);

}

return 0;

}